IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 28 July 2023, accepted 8 August 2023, date of publication 14 August 2023, date of current version 18 August 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3305268

RESEARCH ARTICLE

Virtual Machine Migration Techniques for Optimizing Energy Consumption in Cloud Data Centers

ZHOUJUN MA, DI MA, MENGJIE LV, AND YUTONG LIU
State Grid Jiangsu Electric Power Company Ltd., Nanjing Power Supply Branch, Nanjing 210019, China

Corresponding author: Yutong Liu (yutongliu0110@gmail.com)

This work was supported by the Science and Technology Project of State Grid Corporation of Jiangsu Electric Power Research on ''Key technologies of collaborative regulation of cloud data center resources for cyber-physical systems'' under Grant J2022029.

ABSTRACT The energy used by cloud data centers (CDCs) to support large volumes of data storage and computation is dramatically increasing as the scope of cloud services continues to expand. This puts a greater burden on the environment and results in higher expenses for cloud providers. Virtualization migration and consolidation have been widely used in current CDCs to achieve service consolidation and reduce energy consumption (EC). This study divides the fundamental tasks of virtual machine (VM) migration into three portions: determining migration timing, choosing the VMs to migrate out, and selecting the migration destination hosts. An EC levels-based adaptive dynamic threshold method for determining migration timing was proposed, as well as a correlation and utilization-based strategy for selecting the VMs to migrate out and an improved EC-aware best-fit algorithm for selecting the migration destination hosts. The pro-posed algorithms were evaluated using the CloudSim toolbox, and the real VM workload traces from PlanetLab were used as experimental data. According to the experiments, the proposed algorithms reduce EC, service level agreement violation (SLAV), and the number of VM migrations by an average of 15.49%, 7.85%, and 83.32% in comparison to the related state-of-the-art methods and benchmark algorithms. This suggests that the proposed methods outperform other techniques for VM migration, even when the workload necessitates a significant number of VMs or a greater amount of host resources, and improve the quality of service while optimizing energy consumption. However, the experiments were conducted in a simulation platform, which has some drawbacks, leading to the experimental results varying slightly from the actual environment.

INDEX TERMS Energy consumption optimization, virtual machine migration techniques, dynamic threshold, virtual machine selection, host selection, cloud data center.

I. INTRODUCTION

Cloud computing technology has become increasingly popular and widely used in recent years. This has resulted in the rapid expansion of cloud data centers (CDCs) to meet large-scale data storage and computing demands. However, the expansion of CDCs has led to a rise in the energy consumption (EC) of servers, creating a major challenge for large-scale infrastructures like clusters, grids, and CDCs composed of thousands of heterogeneous servers [1], [2]. The rise in EC has also resulted in high costs for CDCs, reduced profits for cloud providers, and increased CO_2 emissions. Consequently, researchers and industries are now exploring ways to use electricity more efficiently and sustainably to address these challenges.

Servers are usually configured and deployed in CDCs to handle peak workloads and achieve optimal performance. However, this can result in insufficient use of servers during non-peak periods, resulting in resource waste and higher EC. To address this issue, virtualization consolidation has become a common practice in modern data centers (DCs), enabling service consolidation and reducing energy usage through resource multiplexing [3], [4], [5]. By utilizing virtualization technology to create multiple VM instances on a

The associate editor coordinating the review of this manuscript and approving it for publication was Nitin Gupta.

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License.
For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

single physical server, cloud providers can enhance resource utilization and boost their return on investment.

According to recent statistics, servers in an idle state can consume between 50%–70% of the energy used by servers operating at full capacity [6]. This implies that idle servers consume a significant amount of energy without contributing to computing tasks. Consequently, many DCs have implemented energy management schemes that monitor server utilization and trigger VM migration to consolidate underutilized servers. Based on their resource requirements, VMs are reallocated through live migration to minimize the number of active hosts [7]. The power consumption of idle hosts is eliminated by switching them to low-power modes like sleep and hibernation, which further minimize EC. When workload demands increase, the hosts are reactivated to prevent application performance degradation due to resource scarcity. This approach has two primary objectives: minimizing energy consumption and maximizing quality of service (QoS), with QoS requirements determined by the service level agreement violation (SLAV) indicator. Besides EC optimization scenarios, VM migration techniques are used in load balancing, server upgrades, and machine downtime maintenance [8], [9], [10], [11].

This study focuses on the critical tasks involved in VM consolidation migration, namely determining migration timing, selecting the VM to migrate out, and finding suitable migration destination hosts. The methods for addressing these tasks have been analyzed and several limitations have been identified.

To begin with, the study examines two methods for determining migration timing: static and dynamic threshold. The static threshold method, although relatively easy to implement, lacks flexibility as it remains fixed throughout the migration process. On the other hand, the dynamic threshold method is more adaptable and flexible. However, most existing studies primarily consider resource and load situations when setting the migration thresholds, without taking into account EC levels.

Regarding the selection of VMs to migrate out, various migration metrics such as resource utilization, load balancing, migration time, migrated data volume, application performance, and EC are considered. However, there is a need for a method that comprehensively addresses these metrics and incorporates EC considerations.

The study also explores different approaches for finding suitable migration destination hosts, including heuristic, metaheuristic, and machine learning algorithms. However, each of these methods has its limitations. For instance, heuristic algorithms can get trapped in local optima, metaheuristic algorithms often converge slowly, and machine learning algorithms suffer from poor interpretability and a heavy reliance on training data.

To overcome these limitations, this study proposes several improvements and novel approaches. Firstly, an adaptive dynamic threshold method is introduced, which adjusts the migration trigger threshold based on EC levels, resource situations, and data center conditions. This dynamic approach enhances adaptability, optimizes EC, and reduces unnecessary migrations more effectively compared to existing methods.

Additionally, a correlation and utilization-based strategy is proposed for selecting the VMs to migrate out. This strategy prioritizes computing tasks with high load correlation and small comprehensive load values for migration. By doing so, it enables quick restoration of the host to a normal load state, minimizes performance loss during migration, and reduces migration costs.

Furthermore, the study presents an improved EC-aware best-fit algorithm, which has simplicity in implementation and fast convergence. This algorithm contributes to EC reduction and enhances load balancing to a certain extent, leading to improved resource efficiency.

In summary, the main contributions of this study are as follows:

- An adaptive dynamic threshold method based on EC levels is proposed to determine the migration timing, which dynamically and automatically adjusts the migration trigger threshold according to different EC levels and can accelerate the dynamic migration adjustment process of VMs. Additionally, it helps to avoid unnecessary migration. Compared to existing methods, the proposed method is more effective in EC optimization and is better at reducing the number of VM migrations.
- A correlation and utilization-based strategy is proposed for selecting the VMs to migrate out. The computing tasks with high load correlation and small comprehensive load values are selected for migration. This helps to quickly restore the host to a normal load state. Additionally, it has the benefit of minimizing the performance loss of the application during the migration process and reducing the migration cost.
- An improved EC-aware best-fit algorithm with simple implementation and fast convergence is proposed. This helps to reduce EC and improves load balancing to some extent.

By addressing the limitations of existing methods and introducing these novel approaches, this study aims to enhance the efficiency, adaptability, and cost-effectiveness of VM consolidation migration.

The rest of this paper is organized as follows. Section II discusses related work. Section III presents the VM migration scheduling problem formulation and provides the solution. Section IV provides experimental simulation results. Finally, Section V concludes the paper with directions for future work.

II. RELATED WORK

The optimization of EC in CDC has long been a focus of research in the field of information technology. This can be achieved by proposing a reasonable migration strategy while ensuring the QoS. Existing work in EC optimization of CDC can be classified into three aspects: determining migration

timing, selecting migrated VMs, and selecting migration destination hosts. The state of the art of the three aspects is as follows.

A. DETERMINING MIGRATION TIMING

Currently, migration trigger thresholds are defined to decide the time of migration. When the host load rises beyond the overload threshold or falls below the low load threshold, migration is initiated. While moving VMs from underused servers and putting them in sleep mode can assist prevent energy consumption, doing so can also help lower SLAV. Unreasonable migration thresholds, however, might cause a lot of consolidation and frequent, pointless migrations, which can have a bad effect on the performance of the application because of additional delays like migration time and downtime [12]. The service level agreement (SLA) may be broken due to this decline in service quality, which could lead to fines. To reduce the number of VM migrations, it is therefore required to identify overloaded and underloaded hosts. Effective solutions are also required to choose the right time for VM migration.

Migration trigger thresholds can be either static or dynamic. Static thresholds are manually set and remain fixed throughout the entire migration process. Liang et al. [13] and Liu et al. [14] have explored the static threshold approach. In contrast, dynamic thresholds change dynamically based on the current resource situation, cluster load, and time, making them more adaptable and flexible. Beloglazov and Buyya [15], Yadav et al. [6], Singh and Kumar [16], and Kulshrestha and Patel [17] have employed dynamic threshold approaches. Beloglazov proposed two adaptive threshold methods, interquartile range (IQR) and median absolute deviation (MAD), based on a statistical analysis of host historical data. To assess the host's state over time, the present utilization was compared to a dynamically determined threshold. Yadav et al. [6] introduced the GradCent algorithm to calculate the upper limit threshold of a central processing unit (CPU) utilization based on historical CPU workload data and adjust it dynamically. Singh and Kumar [16] proposed a dynamic threshold with enhanced search (DT-ESAR) for VM consolidation systems. Kulshrestha and Patel [17] optimized host overload detection by proposing an exponentially weighted moving average-based threshold formulation method. Host load status is monitored in real-time through load detection procedures, and load prediction methods are used to evaluate the host load state. Load prediction methods predict server load by creating mathematical models, and CDCs can proactively allocate and schedule resources based on the prediction results [18].

The static threshold method for overload and underload state is easy to implement but remains fixed throughout the migration process, limiting flexibility. However, the dynamic threshold varies dynamically with time, resource availability, and cluster load. This approach offers better adaptability and flexibility, but most studies focus on setting the thresholds based on resource and load situations without considering EC levels. The load prediction method can proactively allocate and schedule resources in advance, but they require highly accurate prediction models. Otherwise, inaccurate predictions can lead to additional overhead and worse migration results.

B. SELECTION OF MIGRATED VMS

The VM migration selection algorithm mainly considers one or more migration metrics such as resource utilization, load balancing, migration time, migrated data volume, application performance, and Reddy et al. [19] mainly considered resource utilization and migration time metrics and proposed a VM selection algorithm based on memory utilization, bandwidth utilization, and VM size to optimize the current allocation. Ahmadi et al. [20] considered migration time, migration risk, VM connectivity, freeable resources, and SLAV rate for the selection process and proposed a multi-criteria decision-making method based on hierarchical analysis. Baskaran [21] applied the fuzzy soft set method to select the appropriate VM for migration, which considered CPU usage, memory usage, RAM usage, and correlation values. Mekala and Viswanathan [22] proposed an energy-efficient resource ranking and utilization factor-based VM selection (ERVS) method, which focused on EC and resource utilization metrics. Li et al. [23] selected the out-migration VMs using the content similarity between the VM memories, which can reduce the time of migration, amount of data transmitted, and pressure on network traffic. Haghshenas and Mohammadi [24] proposed a regression-based approach to predict the resource utilization of VMs based on historical data and selected the VMs with higher utilization prediction results for migration.

C. SELECTION OF MIGRATION DESTINATION HOSTS

The selection of the destination host requires choosing the hosts from many hosts to migrate the selected VM, and this task belongs to the nondeterministic polynomial (NP)-hard problem. When the scale of the problem increases, the computational complexity will increase exponentially. Currently, researchers mainly use heuristic, meta-heuristic, and machine learning algorithms to determine the target physical hosts.

Heuristic algorithms mainly include next fit (NF), first fit (FF), best fit (BF), first fit decreasing (FFD), and best fit decreasing (BFD) algorithms. Liang et al. [13] used the BF algorithm to select a target host with the smallest remaining space that can accommodate VMs. Chhikara et al. [25] implemented the BF algorithm using heap structure to determine the target host for the migration container. This implementation has a time complexity of O (1). Chen et al. [26] used the FF, BF, and random algorithms to find a destination host for containers and compare the performance of the three algorithms. Fan et al. [27] used the FF algorithm to ensure that the destination server has enough available resources. Assigning each VM to the host with the smallest increase in power consumption as a result of that assignment is how Beloglazov and Buyya [15] utilize the BFD method to sort all VMs in descending order by their present CPU use.

Meta-heuristic algorithms mainly include simulated annealing, particle swarm optimization, differential evolution, genetic algorithm (GA), ant colony optimization algorithm, and cuckoo search algorithms. Moreover, to jointly exploit the advantages of several metaheuristics, multi-method-based approaches have gained wider attention in the practical field. Liu et al. [14] proposed an algorithm for VM consolidation in CDCs based on ant colony systems and extreme learning machines (ELM). Luo et al. [28] used the improved shuffled frog leaping and improved extreme value optimization algorithms to solve the dynamic allocation problem of VMs and reduce power consumption while satisfying the QoS. He et al. [29] used GA for VM consolidation. Bibiks et al. [30] proposed an improved discrete cuckoo search to solve the resource-constrained scheduling problem. Moazeni et al. [31] proposes a dynamic resource allocation strategy using an adaptive multi-objective teaching-learning based optimization (AMO-TLBO). AMO-TLBO introduces the concept of the number of teachers, adaptive teaching factor, tutorial training, and self-motivated learning. The objectives of AMO-TLBO include minimizing makespan, cost and maximizing utilization using a well-balanced load across virtual machines.

The machine learning algorithms used to determine the target hosts mainly include clustering, classification, reinforcement learning, and neural network methods. Liang et al. [32] proposed a dynamic hybrid machine learning-based algorithm for energy-aware resource deployment in CDCs. They used an extended K-means clustering algorithm and an extended k-nearest neighbors classification algorithm to complete the VM deployment. Ma et al. [10] proposed an online VM scheduling scheme (OSEC) for joint EC and cost optimization based on reinforcement learning theory. Rezakhani et al. [33] applied reinforcement learning and an artificial neural network to propose an integrated algorithm based on energy-aware QoS to dynamically manage VMs in CDCs. Liu et al. [14] proposed a deep reinforcement learning model based on QoS feature learning to optimize DC resource scheduling.

The heuristic algorithm is advantageous because of its simple implementation and fast convergence. However, it is prone to get stuck in local optima. On the other hand, the metaheuristic algorithm can better find the global optimal solution, but it is faced with some limitations, such as slow convergence, excessive parameterization, poor computational result reusability, and difficulty in efficient parameter tuning. Machine learning algorithms have self-learning and self-adaptation capabilities, with superior global search abilities. Nonetheless, these methods suffer from poor interpretability and high reliance on training data.

III. PROBLEM MODEL AND SOLUTION METHODOLOGY

A. EC MODEL AND RESOURCE MODEL OF CDC

The CDC–EC model used in this study will be introduced in this subsection. The CDC's overall EC consists of various components such as the power used by the server host, cooling system, network equipment, and other systems (which may include low-power systems like fire, electrical, lighting, and lightning protection). Since the center's overall EC is the server host's EC, this study mainly focuses on the EC optimization of the host. The overall EC of the center was modeled as the sum of the EC of all hosts, while the EC of the DC in the time interval from t1 to t2 was calculated as shown in (1).

$$E = \int_{t1}^{t2} \sum_{h \epsilon H} P_h(t) \tag{1}$$

where E is the EC of the DC, H is the collection of all hosts in the CDC, and $P_h(t)$ is the power consumption of the host h at time t, t ∈ [t1, t2].

Approaches for evaluating energy efficiency in CDCs can be categorized into three main groups: measurement-based methods, simulation-based methods, and analytical modeling-based methods. Among these, measurement-based evaluation stands out as the most accurate approach, providing real-world data on energy consumption. On the other hand, simulation-based methods offer lower accuracy due to the inherent limitations and simplifications of the models used [34].

The power consumption of a host in a CDC is influenced by various factors, including its hardware configuration and processing components such as the CPU, memory, hard disk, I/O, and network. Previous studies have predominantly relied on linear or square regression models that utilize CPU utilization as the primary parameter to estimate power consumption [35], [36]. However, such an approach fails to consider the impact of memory consumption, which has become increasingly significant with the rise of modern servers equipped with larger memory capacities. Ignoring the power consumption caused by memory can lead to inaccurate estimations. Thus, accurately modeling power consumption for multi-core CPUs is a complex and challenging task.

In light of these challenges, instead of relying solely on specific quantitative models for server power consumption, the study leverages actual power consumption data generated by the SPECpower benchmark results.

The SPECpower committee has launched an energy efficiency benchmark suite called SPECpower_ssj2008. This suite measures the power consumption of a server while running at maximum workload, which is taken as 100%. The workload is divided into 11 discrete zones of 10% each, ranging from 0 to 100%. To determine the dynamic power consumption between two load levels, linear interpolation is used.

HP ProLiant ML110 G4 (Intel Xeon 3040, 2 core, 1860 MHz, 4 GB) and HP ProLiant ML110 G5 (Intel Xeon 3075, 2 core, 2660 MHz, 4 GB) were chosen as the hosts. Table 1 presents the power consumption properties of the selected server. As can be seen, the power consumption of servers increases with higher processor utilization. Also, servers with different CPU frequencies show various power consumption at the same CPU utilization efficiency.

TABLE 1. Power consumption for different level of utilization.

Machine Type	Idle	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
HP G4(Watt)	86.0	89.4	92.6	96.0	99.5	102.0	106.0	108.0	112.0	114.0	117.0
HP G5(Watt)	93.7	97.0	101.0	105.0	110.0	116.0	121.0	125.0	129.0	133.0	135.0

Then, the resources of the CDC were described, mainly including the server host and VM sets. The host was taken as the resource provider, while the VM was the resource demander. The details are expressed in (2) and (3).

$$P_{i,d} = \left(P_{i,cpu}, P_{i,mem}, P_{i,bw}, \ldots\right) \tag{2}$$

$$V_{j,d} = \left(V_{j,cpu}, V_{j,mem}, V_{j,bw}, \ldots\right) \tag{3}$$

Formula (2) describes the d-dimensional resources available on host i, where $P_{i,cpu}$, $P_{i,mem}$, and $P_{i,bw}$ are the CPU, memory, and bandwidth resources owned by host i, respectively, whereas Formula (3) represents the d-dimensional resource requirements of VM j, where $V_{j,cpu}$, $V_{j,mem}$ and $V_{j,bw}$ represents the CPU, the memory, and the bandwidth resource demand of VM j, respectively.

B. DETERMINE THE TIMING OF MIGRATION: ADAPTIVE DYNAMIC THRESHOLD BASED ON EC LEVELS

Migrating VMs from overloaded hosts helps reduce SLAV, while migrating VMs from less-efficient servers and switching them to energy-efficient mode reduces EC. The migration timing in this study was determined by setting a migration trigger threshold. This was because unreasonable migration thresholds may result in excessive consolidation and frequent meaningless migration, further leading to decreased QoS and other negative effects.

An adaptive dynamic threshold method was proposed based on EC levels (ADT-EC), which dynamically and automatically adjusts the migration trigger threshold according to different EC levels. This method will help to avoid unnecessary migration while reducing EC optimization time.

First, the EC level was defined within the CDC, as expressed in (4).

$$Level = \begin{cases} BLUE, & E < aE_{max} \\ GREEN, & aE_{max} \le E \le bE_{max} \\ RED, & E > bE_{max} \end{cases} \tag{4}$$

where E_{max} is the maximum allowable EC of the CDC and a and b are the adjustable coefficients that satisfy that $0 < a < b < 1$. The EC level is BLUE when the center's EC is lower than aE_{max}, indicating an ideal state. When the EC is between aE_{max} and bE_{max}, the level is GREEN, indicating a normal state. When EC is larger than bE_{max}, the EC level is RED, indicating an undesirable state. The overload threshold is calculated using (5), which is given as follows.

$$thr_{high} = \begin{cases} 0.9, & BLUE \\ max\left\{\left(thr_{high} - step\right), 0.7\right\}, & GREEN \\ max\left\{\left(thr_{high} \times \alpha\right), 0.6\right\}, & RED \end{cases} \tag{5}$$

Algorithm 1 ADT-EC

Input: HostList and EC of CDC
Output: OverutilizedHostList and UnderutilizedHostList
1: **for each** host in HostList **do**
2: count←0
3: utilizationHistory←getHostUtilizationHistory(host, T)
//Obtain the m load data of the host in the time interval T
4: level←getECLevel(EC of CDC)
//Obtain the CDC's EC level by (4)
5: thr_{high} ←updateUtilizationThreshold(level)
//Update the threshold by (5)
6: **for each** i=1 to m **do** //m is same as line 3
7: **if** utilizationHistory[i] > thr_{high}, **then**
8: count++
9: **if** utilizationHistory[i] < thr_{low}, **then**
10: add host to UnderutilizedHostList
11: break
12: **if** count > n, **then** //n < m
13: add host to OverutilizedHostList
14: **return** OverutilizedHostList and UnderutilizedHostList

where the step for adjustable parameters ranges from 0 to 0.1, and the values of α for adjustable parameters range from 0 to 1. When the EC level is in the BLUE state, thr_{high} is set to 0.9. When the EC level is in the GREEN state, thr_{high} enters the fixed step length reduction process. When the EC level is in the RED state, the thr_{high} enters the process of fixed coefficient reduction. The underload threshold $thr_{low} = 0.3$ remains constant.

The server utilization may occasionally exceed the maximum threshold for a short time, dropping the load rapidly. There is no need to move the VM from this server in order to free up resources because the host is not overloaded in this instance. Therefore, it is essential to prevent unnecessary migration. In the time interval T, when at least n of m load data is higher than thr_{high} (n < m), it is possible that the host is in the overloaded state and will trigger the overload migration. Unlike overload migration, because the underload threshold is set low enough when the load data is below the thr_{low}, it is possible that the host is in the underload state and can trigger the underload migration immediately.

Algorithm 1 describes the pseudo-code of the ADT-EC, which helps to understand the entire workflow of this algorithm.

Algorithm 1 inputs the host list and EC of CDC to get the overloaded and underloaded host list. First, the load data of

the host in the time interval T (line 3) is obtained, then the load status of the host is judged, and the dynamic threshold (lines 4–5) is updated before performing the judgment. To prevent unnecessary migration, when at least n load data is higher than thr_{high} (lines 7–8), the host is added to the overloaded host list (lines 12–13). However, when the load data is below the thr_{low}, the host is immediately added to the underloaded host list (lines 9–11).

C. SELECT THE VM TO MIGRATE: SELECTION STRATEGY BASED ON CORRELATION AND UTILIZATION

It is necessary to effectively select the VM that moves out when the server host is overloaded, minimize the negative impact on the migration process, and quickly restore the host's normal load state. Therefore, the selection strategy based on correlation and utilization (SS-CAU) in this study was proposed, and the SS-CAU method is described as follows.

The SS-CAU selects the VM from the set of VMs with the least CPU utilization and high load correlation to migrate. Selecting the VM with a high correlation can accelerate the host restore to the normal load state because the higher the correlation between the VM load and the host load, the greater the possibility of incurring the host overload. Therefore, under the premise of high correlation, the VM with the least CPU utilization was selected. This method has the benefit of minimizing the performance loss of the application during the migration process and reducing the migration cost.

Regression analysis was used to calculate the load correlation where the load sequence of the VM is $\{x_1, x_2, \ldots, x_n\}$ and the load sequence of the host is $\{y_1, y_2, \ldots, y_n\}$. The correlation coefficient is expressed as (6).

$$R^2 = \frac{\sum_{i=1}^{n} (x_i - \bar{x})^2 (y_i - \bar{y})^2}{\sum_{i=1}^{n} (x_i - \bar{x})^2 \sum_{i=1}^{n} (y_i - \bar{y})^2} \tag{6}$$

The load correlation between the VM and host is denoted by R^2, where $\bar{x}$ and $\bar{y}$ are the average value of the load data for the VM and host, respectively. The value of R^2 ranges between 0 and 1, with a higher value indicating a stronger correlation between the VM and the host. When $0.6 < R^2 \leq 1$, it indicates a strong correlation between the VM and the host; the VM that meets this criterion is added to the candidate set. Then, the VM with the lowest CPU utilization is chosen from the set and migrated to another host until the overloaded host returns to its normal load level.

Algorithm 2 describes the pseudo-code of the SS-CAU.

Algorithm 2 inputs the overloaded hosts and outputs the selected VMs to be migrated. First, the list of VMs on the overloaded host (line 2) is obtained, then the correlation coefficient of the VM (lines 4–5) is calculated according to Formula (6). The VMs with a correlation coefficient greater than 0.6 are added to the candidate set (lines 6–7), which is then sorted in ascending order by utilization (line 8). Finally, the VMs are moved out in sequence until the overloaded host returns to the normal load level (lines 9–12).

Algorithm 2 SS-CAU

Input: OverutilizedHostList
Output: SelectedVMList //Selected VM which will be migrated
1: **for each** host in OverutilizedHostList **do**
2: vmList←getMigratableVms(host) //Get the list of VMs on the overloaded host
3: candidateVmList←null //The VM that meets the strong correlation criterion will add to the candidate list.
4: **for each** vm in vmList **do**
5: r←getCorrelationCoefficient(vm, host) //The correlation coefficient of vm is calculated according to (6)
6: **if** r > 0.6, **then**
7: add vm to candidateVmList
8: sortedVMList←sortByUtilization(candidateVmList) //Sort candidateVmList in ascending order by utilization
9: **for each** vm in sortedVMList **do**
10: add vm to SelectedVMList
11: **if** the host returns to the normal load level, **then**
12: break
13: **return** SelectedVMList

D. SELECT THE DESTINATION HOSTS: IMPROVED ENERGY-AWARE BEST-FIT ALGORITHM

Selecting the destination hosts is essentially similar to the initialization placement problem of the VM. Therefore, it is necessary to map the VM to the appropriate host. The host not only needs to meet the VM's resource requirements but also consider saving energy, improving load balance, and resource utilization. The task of selecting the destination host is abstracted as the optimization problem, as expressed in (7).

$$\begin{aligned} &min\,(E_{DC} = \sum_{i=1}^{n} E_{PMi}) \\ &st.\,VM_{request}\,(CPU, MEM, BW, Disk) \\ &\quad \leq PM_{surplus}\,(CPU, MEM, BW, Disk) \\ &\quad U_{low} < PM_{utilization} < U_{high} \end{aligned} \tag{7}$$

Our objective is to minimize EC in the CDC. The first constraint requires that the resources provided by the selected host be greater than the migrated VM's requests. The second constraint ensures that the selected host's resource utilization remains within the ideal interval to prevent frequent migrations due to high or low utilization. Thereby reducing the number of VM migrations, minimizing performance losses, and improving load equilibrium and resource utilization.

Unfortunately, this is an NP-hard problem, and the computational complexity grows exponentially as the problem scale expands. The heuristic greedy algorithm was adopted to avoid high overhead, and an improved energy-aware best-fit (IEABF) algorithm was proposed based on Beloglazov's work [7].

Algorithm 3 provides the pseudo-code for the IEABF.

Algorithm 3 IEABF

Input: HostList and SelectedVMList //The list of all hosts in CDC and the list of VM which will be migrated
Output: SelectedHostList //The list of the destination hosts
1: sort SelectedVMList in decreasing order by CPU utilization
2: **for each** vm in SelectedVMList **do**
3: minEnergy←Max //Assign a max value to the metric of energy
4: selectedHost←null
5: excludedHosts←ADT-EC(HostList) //Overload and underload hosts are excluded using algorithm 1
6: **for each** host in HostList **do**
7: **if** excludedHosts.contains(host), **then**
8: continue
9: **if** host.isSuitableForVm(vm), **then** //The host meets the VM resource requirements
10: energy←getEnergyAfterPlacement(vm, host)
11: **if** energy < minEnergy and U_{low} < host.utilization < U_{high}, **then**
12: minEnergy←energy
13: selectedHost←host
14: **if** selectedHost is not null, **then**
15: add selectedHost to SelectedHostList
16: **return** SelectedHostList

Algorithm 3 takes the list of all hosts in CDC and outgoing VMs as input and maps each VM to an appropriate host. First, the VMs are sorted in descending order by CPU utilization (line 1). This is conducted by giving higher priority to VMs with high CPU utilization. The algorithm excludes overloaded and underloaded hosts from the list for each VM using algorithm 1 (lines 5–8). From the remaining host list, the algorithm selects the host that meets the VM's resource requirements while consuming the lowest energy (lines 9–13). Additionally, the algorithm ensures that the selected host's resource utilization falls within the ideal range (line 11). After the loop, the algorithm returns the list of the destination hosts.

IV. EXPERIMENTAL EVALUATION

A. EXPERIMENTAL SETUP

In this study, the CloudSim toolbox was used to conduct the simulation experiments of the proposed algorithm. CloudSim is a widely used cloud computing simulation platform software that provides DC-based virtualization technology, virtual cloud modeling, and simulation functions. Each entity in CloudSim is a simulated instance of CDC components, including CDC, host, VM, agent, and cloud task. Both the host and VM have corresponding computing capabilities.

A CDC with 800 heterogeneous servers was developed for this study: 400 HP ProLiant ML110 G4 dual-core machines, each 1860 MIPS; 400 HP ProLiant ML110 G5 dual-core machines, each 2660 MIPS. Both servers have 4 GB memory and support 1 GB/s bandwidth. According to the SPECpower benchmark, Table 1 in Section III presents the power consumption characteristics of these servers, while Table 2 presents other characteristics.

TABLE 2. Server characteristic parameters.

Server	CPU (MIPS)	core	Mem (GB)	BW (GB/s)
G4	1860	2	4	1
G5	2660	2	4	1

TABLE 3. Vm characteristic parameters.

VM Instance Type	CPU (MIPS)	RAM (GB)
High-CPU medium	2500	0.850
Extra-large	2000	3.750
Small	1000	1.700
Micro	500	0.613

TABLE 4. The characteristics of workload dataset.

Date	No. of VMs	Mean (%)	SD (%)
03-03-2011	1052	12.31	17.09
06-03-2011	898	11.44	16.83
09-03-2011	1061	10.70	15.57
22-03-2011	1516	9.26	12.78
25-03-2011	1078	10.56	14.14
03-04-2011	1463	12.39	16.55
09-04-2011	1358	11.12	15.09
11-04-2011	1233	11.56	15.07
12-04-2011	1054	11.54	15.15
20-04-2011	1033	10.43	15.21

VM instances include high-memory and High-CPU, while VM sizes include large, medium, small, and micro. Four Amazon EC2 VMs were used in this experiment. Table 3 presents the details of the VM instances. Additionally, the start/stop delay of the VM directly affects the SLAV index during the experiment. Therefore, the start/stop delay of the VM was set to 100 s.

To show the accuracy and practicability of the proposed methods, experiments were conducted using real workload data provided by the CoMon project, which is a monitoring infrastructure of PlanetLab. This dataset includes bandwidth, CPU utilization, and memory usage of over 1000 hosts in 500 locations worldwide. The workload data was collected between March 3 and April 20, 2011. These workloads cover a range of VM numbers and different resource utilization characteristics, such as average CPU utilization and standard deviation. Each VM comprises 288 CPU utilization records, measured every 5 min. These data were then interpolated to generate CPU utilization per second. Table 4 presents the characteristics of workload dataset.

TABLE 5. Experimental design.

Experiments	Evaluation Objectives	Comparison Methods	Others
Experiment 1	ADT-EC	THR_0.8 MAD_2.5 IQR_1.5 LR_1.2 LAOD MDP_3.0 EPA	MMT for selecting the VMs to migrate out PABFD for selecting the destination hosts
Experiment 2	SS-CAU	MMT MU MPCM MUMA	LR for determining migration timing PABFD for selecting the destination hosts
Experiment 3	IEABF	PABFD TPSA ALBA HPNBFD SABFD AntPu AntVc	LR for determining migration timing MMT for selecting the VMs to migrate out
Experiment 4	ADT-EC_ SS-CAU_ IEABF	THR_0.8_MMT_PABFD LAOD_MMT_PABFD LR_1.2_MU_PABFD LR_1.2_MUMA_PABFD	/
		DTHMF MMSD_FS EPA_AMLA EQ_DVMCA PPAVP	

B. EVALUATION METRICS

Different indicators were used to compare the effectiveness of the proposed algorithm. The main focus of this study was to minimize the EC of the CDC under the premise of ensuring the QoS. First, the EC of the DC was calculated using (1). Second, SLAV and the number of VM migrations were used as QoS indicators to evaluate the proposed algorithm. The increase in the SLAV indicates that the resource allocation scheme of the migration algorithm is not perfect. It also indicates that the resources in the VM are either insufficient or the number of VMs is insufficient to meet the resource requirements of users. The description of SLAV is expressed in (8).

$$SLAV = SLAVO \times SLAVM \tag{8}$$

SLAV is the combined impact of SLA violation due to overloaded hosts (SLAVO) and SLA violation for migrations (SLAVM). (9) and (10) provide the calculation criteria for SLAVO and SLAVM.

$$SLAVO = \frac{1}{M}\sum_{i=1}^{M}\frac{T_{si}}{T_{ai}} \tag{9}$$

M describes the count of servers, T_{si} is the total time when host i experienced 100% utilization leading to SLA Violation, T_{ai} is the total active time of host i.

$$SLAVM = \frac{1}{N}\sum_{j=1}^{N}\frac{C_{dj}}{C_{rj}} \tag{10}$$

N describes the count of VMs, C_{dj} stands for the CPU request at the time of migration of VM j and C_{rj} stands for total CPU requested by VM i.

The increased number of VM migrations may result in increased system overhead and instability, mainly due to the following:

- VMs are heavyweight, and multiple migrations will cause massive file replication between hosts, consuming bandwidth and congesting the network.
- The migration process will fail or roll back due to packet loss during transmission.
- VM migration will cause long service pauses (long file transfer and start/stop time).

C. EXPERIMENTAL DESIGN

In this section, we design experiments to assess the effectiveness and efficiency of our proposed approaches. Specifically, we need to evaluate the performance of the ADT-EC in determining migration timing, the SS-CAU in selecting VMs to migrate out, and the IEABF in selecting migration destination hosts. To compare our proposed algorithms, we simulated

FIGURE 1. Comparison of average EC for ten workloads – exp1.

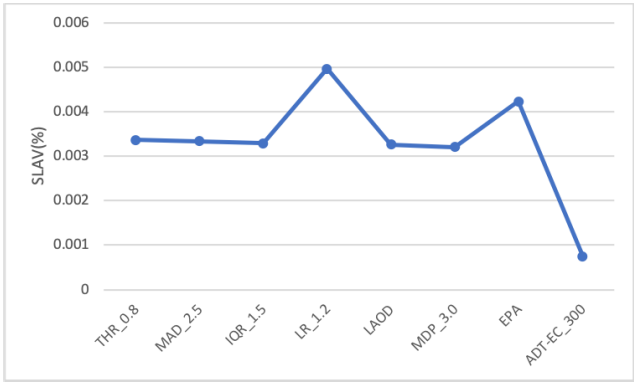

FIGURE 2. Comparison of average SLAV for ten workloads – exp1.

them alongside other threshold methods, VM selection algorithms, and host selection algorithms.

Four questions require investigation:

RQ1: How does the ADT-EC method perform on heterogeneous hosts using real PlanetLab workloads compared to other migration timing methods?

RQ2: How does the SS-CAU perform on heterogeneous hosts using real workloads compared to other VM selection methods?

RQ3: How does the IEABF perform compared to other host selection methods using real workloads?

RQ4: How does the combined effect of the ADT-EC approach, SS-CAU, and IEABF compare to advanced VM migration consolidation strategies?

To address these questions, we designed Experiments 1 to 4 as outlined in Table 5.

D. ANALYSIS OF EXPERIMENTAL RESULTS

1) EXPERIMENT 1 - EVALUATION OF ADT-EC ALGORITHM

For Experiment 1, we simulated and analyzed the performance of the ADT-EC algorithm. As benchmark algorithms, we selected the static threshold (THR), median absolute deviation (MAD), inter-quartile range (IQR), and local regression (LR) methods [7]. Additionally, we compared the results of our proposed algorithm against three advanced algorithms: LAOD [37], MDP_3.0 [38], and EPA [39]. In this experiment, various threshold methods were used for the migration timing determination phase. Minimum migration time (MMT) [15] was used to determine the migration out VM method, while power aware best fit decreasing (PABFD) [15] was used to determine the migration in the destination host method.

- THR: This manually sets the migration threshold, causing it not to change during the migration process. The best static threshold is determined to be 0.8 through research [13].
- MAD: This analyzes the change in the historical load of the host and calculates the median absolute deviation of the CPU utilization to dynamically adjust the overload threshold. The degree of VM consolidation is determined by the experiment's safety parameter, which has a value of 2.5. This parameter controls the safety level, with a lower level resulting in less EC but a greater SLAV due to consolidation.
- IQR: This method is similar to the MAD method in that it calculates the IQR of CPU utilization and reserves additional resources for hosts with unstable loads. The safety parameter is similar to MAD, which has a value of 2.5.
- LR: Use linear regression to predict the CPU utilization of servers in a time series method. It identifies overloaded physical machines at each time interval. The safety parameter is similar to MAD and IQR, which has a value of 1.2.
- LAOD: The method, which is based on learning automata, aims to predict the CPU utilization of VMs to estimate whether a host is overloaded or not. Each VM is equipped with its learning automaton, which can take actions to increase, decrease, or maintain its CPU utilization. The predicted CPU utilization of a host is calculated as the sum of the predicted utilizations of all VMs.
- MDP: The threshold selection is modeled as a Markov decision process. With the solution of the improved Bellman optimality equation by the value iteration method, the optimization model is resolved, and the optimum overload threshold is adaptively selected. MDP-3.0, in which the "3.0" denotes a slide window size of 3u in the online probability estimation for the stat transition probability. The u represents the times of VMs consolidation with one-day workload trace due to conducting a VMs consolidation each 5 min.
- EPA: Calculate the MAD distance criterion for three prediction models: simple exponential smoothing, double exponential smoothing, and polynomial regression. The weights of the forecasting models are determined based on the MAD criterion. The median of the server's CPU utilization history and the forecasting models, along with their weights, are used to predict the server's CPU utilization the next time. The predicted value is then compared to the threshold to determine whether the server is overloaded.

Fig.1–3 show the specific results of the experiment.

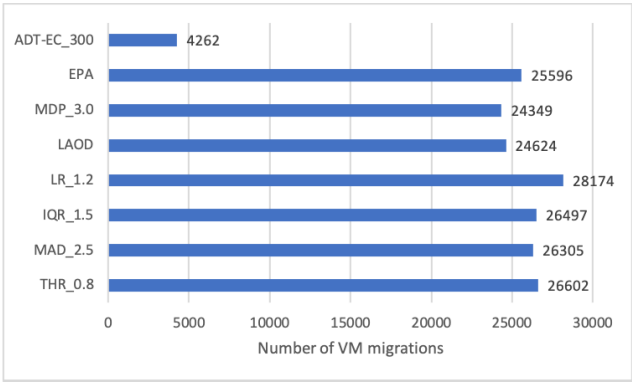

FIGURE 3. Comparison of average number of VM migrations for ten workloads – exp1.

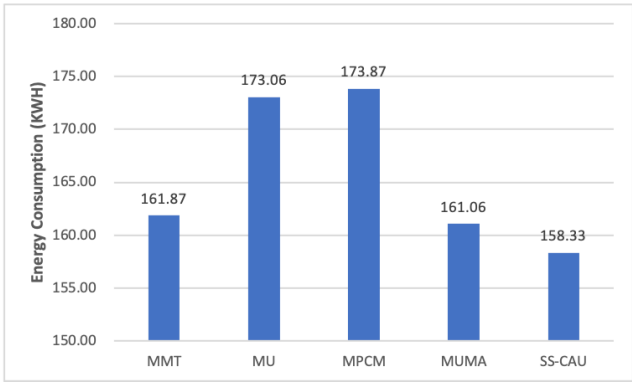

FIGURE 4. Comparison of average EC for ten workloads – exp2.

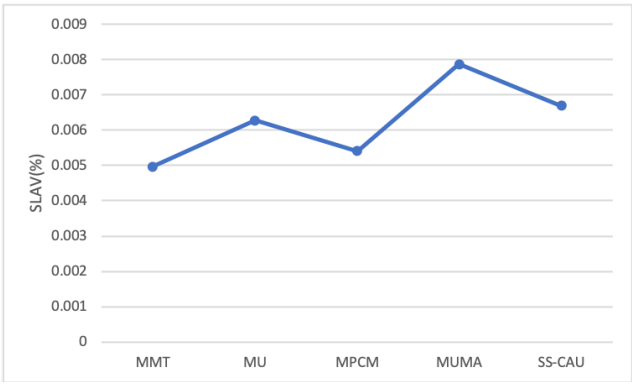

FIGURE 5. Comparison of average SLAV for ten workloads – exp2.

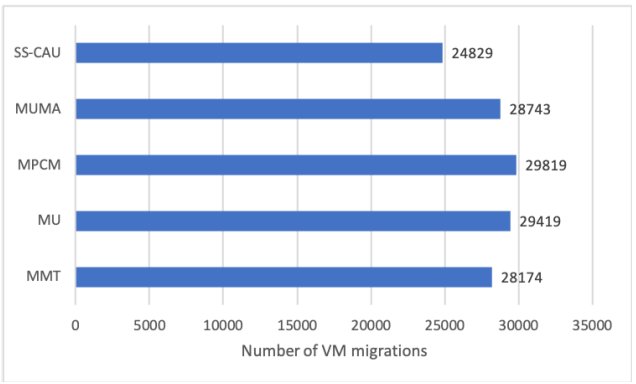

FIGURE 6. Comparison of average number of VM migrations for ten workloads – exp2.

The proposed ADT-EC algorithm outperforms other algorithms in terms of EC, SLAV, and the number of VM migrations, as shown in Fig.1–3. Fig.1 illustrates EC when executing all workload samples under different threshold methods. Compared to THR, MAD, IQR, LR, LAOD, MDP, and EPA, ADT-EC reduces the total EC of the CDC by an average of 23.43%. Fig.2 shows that the SLAV is reduced by an average of 79.04%. Additionally, the ADT-EC approach significantly reduced the number of VM migrations, as shown in Fig.3, the number of VM migrations is reduced by an average of 83.59%. VM migrations require additional system overhead and have a non-negligible migration cost, so minimizing the number of migrations is sensible. These all show that ADT-EC outperforms the other algorithms.

2) EXPERIMENT 2 - EVALUATION OF SS-CAU METHOD

This experiment focused on comparing different methods in selecting the migration out VMs. The performance of the SS-CAU method was assessed and compared with the minimum migration time (MMT) [15], minimum utilization (MU) [15], MPCM [40], and MUMA [41] policy. LR was used for the migration timing determination phase, and PABFD for selecting the destination host phase.

- MMT: The MMT policy migrates a VM that requires the minimum time to complete a migration relative to the other VMs allocated to the host. The migration time is estimated as the amount of RAM utilized by the VM divided by the spare network bandwidth available for the host.
- MU: The VM with the lowest CPU utilization is chosen for migration.
- MPCM: The MPCM method selects VMs that have a minimum product of RAM and CPU utilization.
- MUMA: The MUMA method selects VMs with the highest amount of resource utilization and the lowest amount of allocated resource.

Fig.4–6 show the experimental results.

Fig.4 shows the average EC values for ten workloads under different VM selection policies. Compared to the MMT, MU, MPCM, and MUMA methods, the proposed SS-CAU method reduces EC by an average of 5.33%. Fig.5 shows that the SLAV of SS-CAU is slightly above average but still in the acceptable range.Fig.6 shows that the SS-CAU method reduces the number of VMs migrated by an average of 14.46%.

3) EXPERIMENT 3 - EVALUATION OF IEABF METHOD

Experiment 3 uses the PABFD, TPSA [42], ALBA [43], HPNBFD [44], SABFD [45], AntPu [46], and AntAc [46] as

comparison methods to evaluate the IEABF method. In this experiment, different methods were used for the destination host determination phase. LR and MMT methods were used for the migration timing determination and VM selection phases, respectively.

- PABFD: All VM are sorted according to the descending order of their current CPU utilization. Each VM is allocated to the host with the least increase in power consumption caused by the allocation.
- TPSA: Take advantage of TOPSIS as a multi-criteria algorithm that considers five criteria depicted (power increase, available capacity, number of VMs, resource correlation, and migration delay) in its decision process. This policy computes the scores of all the hosts which candidate for hosting a VM and selects the PM with the highest score.
- ALBA: This method is based on the best-fit decreasing algorithm, which uses learning automata theory, correlation coefficient, and ensemble prediction algorithm in VM allocation. Also, the proposed approach uses two measures to decrease the SLAV: the first one is the adequacy of resources, and the second one is the minimum correlation coefficients between the current VMs and the host VMs.
- HPNBFD: The sorted components at module, rack, and host levels are evaluated hierarchically, starting from the top (module level) and ending at the bottom (host level). If a module is predicted to have a high or low load, it is immediately excluded as a potential destination for the migrating VM during the first evaluation. This process is repeated until a suitable module is found based on its predicted future load.
- SABFD: The VMs selected to migrate are sorted in decreasing order of CPU utilization. The hosts which have enough resources in MIPS will be estimated for the first VM. Then, the host with minimum available MIPS after the VM is placed will be selected to migrate this VM to.
- AntPu: VMs placement was done using the max-min ant system technique. Predicted utilization of host resource is incorporated to design the heuristic information and cost function.
- AntAc: VMs placement was done using the max-min ant system technique. In which available capacity (available capacity is used in most of the Ant-based solutions as heuristic information) of host resource is used to design the heuristic information and cost function.

Fig.7–9 show the experimental results of EC, SLAV, and the number of VM migrations when executing all workload samples under different host selection techniques.

As shown in Fig.7, the proposed IEABF algorithm outperforms existing algorithms by reducing EC by an average of 22.19%. Fig.8 shows that the SLAV is in an average medium position. Additionally, as shown in Fig.9, the proposed IEABF algorithm significantly reduces the number of VM migrations by an average of 66.02%. These results demonstrate that the IEABF algorithm effectively reduces EC while significantly improving the QoS.

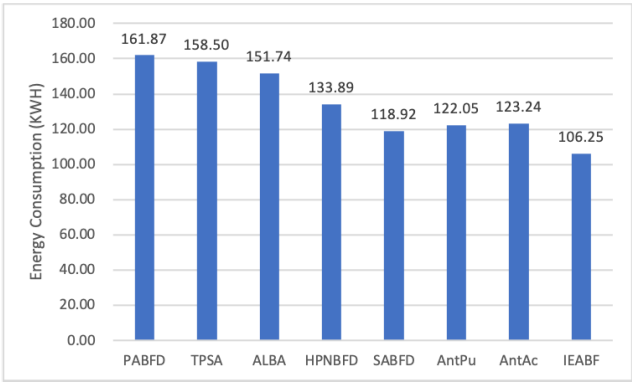

FIGURE 7. Comparison of average EC for ten workloads – exp3.

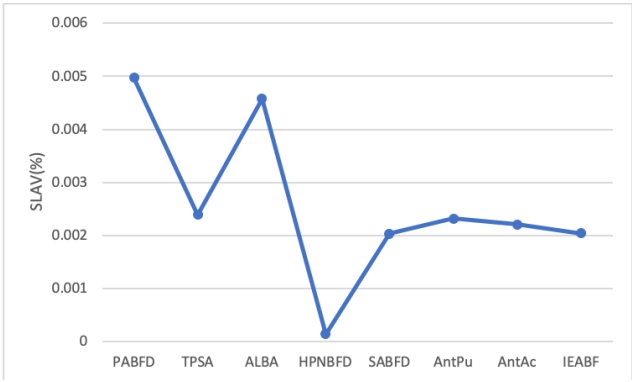

FIGURE 8. Comparison of average SLAV for ten workloads – exp3.

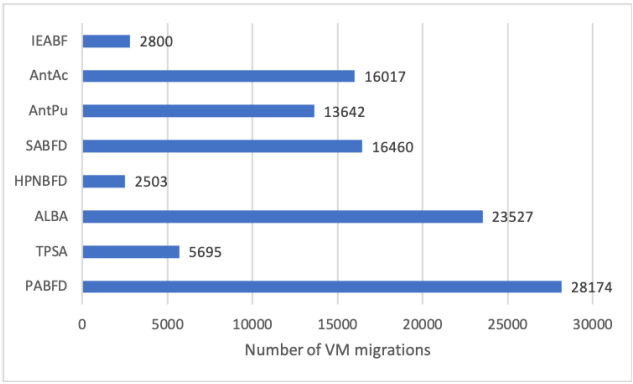

FIGURE 9. Comparison of average number of VM migrations for ten workloads – exp3.

4) EXPERIMENT 4 - COMPREHENSIVE EVALUATION

To evaluate the combined effect of the ADT-EC, SS-CAU, and IEABF algorithms, results are compared with the related state-of-the-art methods, including DTHMF [47], MMSD_FS [48], EPA_AMLA [39], EQ_DVMCA [49], and PPAVP [50], in addition to four baselines: THR_MMT_PABFD, LAOD_MMT_PABFD, LR_MU_PABFD, and LR_MUMA_PABFD.

TABLE 6. The statistical analysis of results.

Experiments	Metrics	EC (KWH)	Number of VM migrations	SLAV (%)
Experiment1	Mean	133.30	4261.60	0.00075
	SD	25.52	575.95	0.00020
	Max	188.79	5272	0.00119
	Min	102.18	3249	0.00051
	Average improvement	23.43%	83.59%	79.04%
Experiment2	Mean	158.33	24829.30	0.00669
	SD	28.21	4555.75	0.00094
	Max	216.16	32679	0.00917
	Min	122.87	17484	0.00572
	Average improvement	5.33%	14.46%	-8.99%
Experiment3	Mean	106.25	2800.80	0.00204
	SD	20.24	340.38	0.00052
	Max	147.80	3421	0.00318
	Min	83.43	2257	0.00142
	Average improvement	22.19%	66.02%	7.88%
Experiment4	Mean	129.96	2841.50	0.00048
	SD	24.43	421.72	0.00012
	Max	180.38	3496	0.00070
	Min	100.39	2238	0.00037
	Average improvement	15.49%	83.32%	78.53%

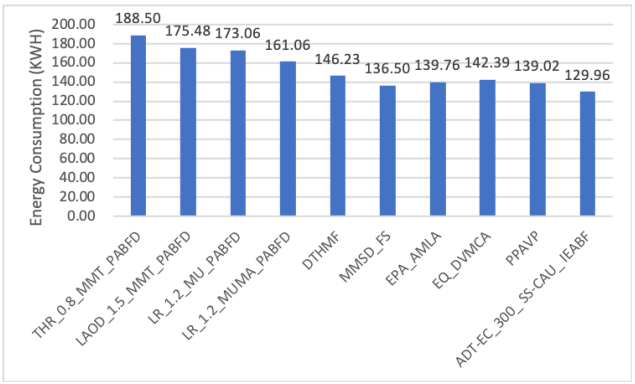

FIGURE 10. Comparison of average EC for ten workloads – exp4.

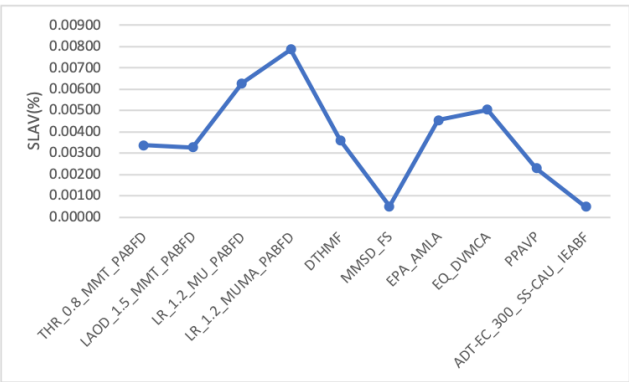

FIGURE 11. Comparison of average SLAV for ten workloads – exp4.

- DTHFM: A combined strategy using the best fit decreasing bin packing method and multi-pass optimization in VM placement for an efficient VM consolidation.
- MMSD_FS: A Fuzzy VM selection method, which incorporates migration control, can enhance the performance of the selection strategy. An overload detection algorithm has also been proposed based on the mean, median, and standard deviation of the utilization of VMs.
- EPA_AMLA: The analysis phase predicts the future workload using an ensemble prediction method composed of simple exponential smoothing, double exponential smoothing, and polynomial regression models to proactively handle workload fluctuation. In the planning phase, utilize the learning automata algorithm as a decision-maker that tunes the weight of the heuristics to obtain the self-optimizing decisions for virtual machine selection.
- EQ_DVMCA: Based on balancing EC and QoS to achieve the efficient consolidation of virtual resources. Propose a hybrid load detection algorithm for determining migration timing. Propose a VM selection algorithm based on CPU and memory perception. Propose a VM placement algorithm based on resource-demand scaling.
- PPAVP: The penalty-aware and cost-efficient method considers cloud resource management as a cost problem. In this method, parameters such as user budget, penalty, and host energy consumption cost play an important role in minimizing operational cost which leads to higher profit for cloud providers.

Fig.10–12 show the experimental results.

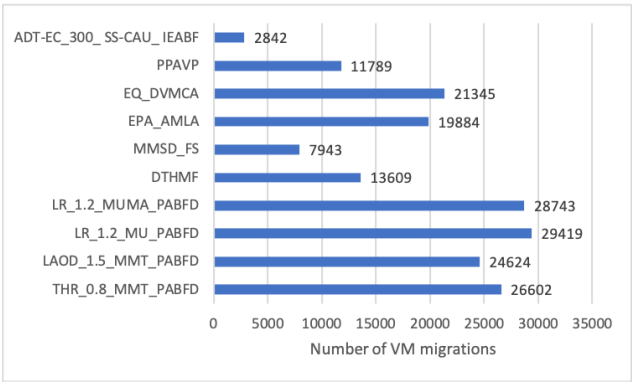

FIGURE 12. Comparison of average number of VM migrations for ten workloads – exp4.

The proposed algorithms reduced the EC by an average of 15.49% (Fig.10), the SLAV by an average of 7.85% (Fig.11), and the number of VM migrations by an average of 83.32% (Fig.12) in comparison to the related state-of-the-art methods and benchmark algorithms. These results show that the proposed methods outperform other techniques for VM migration, even when the workload necessitates a significant number of VMs or a greater amount of CPU resources.

E. DISCUSSION

In this discussion sub-section, we will analyze and discuss the results of our work, highlighting the advantages of our approach and acknowledging the limitations of our research.

The results are compared with the state-of-the-art methods. In the comparison, statistical analysis of results showed the performance improvement of the proposed methods. The details are shown in Table 6.

1) ADVANTAGES

- Determining Migration Timing:

Compared to existing methods, our ADT-EC method demonstrates superior performance in EC optimization and reduces unnecessary VM migrations. By dynamically adjusting the migration trigger threshold based on EC levels, resource situations, and data center conditions, our method achieves better adaptability and more effective EC optimization.

- Selecting VMs to Migrate Out:

The SS-CAU method prioritizes computing tasks with high load correlation and small comprehensive load values for migration. This strategy enables quick restoration of the host to a normal load state, minimizes performance loss during migration, and reduces migration costs. Although the SLAV of SS-CAU was slightly above average, it remained within an acceptable range.

- Selecting Destination Hosts:

The IEABF algorithm contributes to EC reduction and enhances load balancing to some extent, leading to improved resource efficiency. It combines simplicity in implementation with fast convergence, providing an efficient solution for host selection during VM consolidation migration.

- Combined Effect and Comparison:

When evaluating the combined effect of the ADT-EC, SS-CAU, and IEABF algorithms, our proposed methods outperformed related state-of-the-art methods and benchmark algorithms. The combined effect resulted in an average reduction of EC by 15.49%, SLAV by 7.85%, and the number of VM migrations by 83.32%.

These results demonstrate the effectiveness of our proposed approaches in enhancing the efficiency, adaptability, and cost-effectiveness of VM consolidation migration. Our methods show superior performance even when dealing with a significant number of VMs or a greater amount of CPU resources.

2) LIMITATIONS

While our proposed approaches have shown promising results, certain limitations should be acknowledged:

The comparison with existing methods was based on simulated experiments, and the performance in real-world implementations may vary. Nonetheless, the experimental dataset comprehensively covers a wide range of application scenarios, and the representative data it contains can still effectively illustrate the problem at hand.

The proposed approaches primarily focus on VM consolidation migration and may not address other aspects of virtualized environments or cloud computing systems.

V. CONCLUSION

This study focuses on optimizing the EC of CDCs by dividing the VM migration tasks into three parts: determining migration timing, selecting the VMs to migrate out, and finding the destination hosts to migrate in. To accomplish this, three algorithms were proposed: ADT-EC, SS-CAU, and IEABF. Using the CloudSim toolbox, four simulation experiments were conducted.

The experimental results clearly demonstrate that our proposed algorithms outperform the benchmark algorithms in three significant ways. Firstly, they achieve an average reduction in EC of 15.49%, indicating their effectiveness in minimizing energy consumption. Secondly, there is an average reduction in SLAV of 7.85%, highlighting the improvement in meeting the desired QoS. Lastly, the algorithms successfully reduce the number of VM migrations by an average of 83.32%, which is crucial for minimizing system overhead and migration costs. The results of our experiments validate the superiority of our approaches over benchmark algorithms, showcasing their potential for enhancing energy efficiency and overall system performance.

Our future work will focus on the following three aspects:

- VM consolidation migration entails multiple objectives such as load balancing, performance optimization, and cost reduction. Future studies can explore the application of multi-objective optimization techniques to find optimal trade-offs among these objectives.

- To validate the effectiveness and practicality of our proposed approaches, further evaluation in large-scale real-world deployments is necessary. This can provide insights into the challenges and benefits of implementing these approaches in production environments.
- As container technology is becoming more popular for resource virtualization, we plan to enhance the migration algorithm developed in this study to make it compatible with containers.

APPENDIX

Code and data are available from https://github.com/YutongLiu0110/VMC-for-Optimizing-Energy.

REFERENCES

[1] M. Zakarya, "Energy, performance and cost efficient datacenters: A survey," *Renew. Sustain. Energy Rev.*, vol. 94, pp. 363–385, Oct. 2018, doi: 10.1016/j.rser.2018.06.005.

[2] M. H. Shirvani, A. M. Rahmani, and A. Sahafi, "A survey study on virtual machine migration and server consolidation techniques in DVFS-enabled cloud datacenter: Taxonomy and challenges," *J. King Saud Univ.-Comput. Inf. Sci.*, vol. 32, no. 3, pp. 267–286, Mar. 2020, doi: 10.1016/j.jksuci.2018.07.001.

[3] N. Hamdi and W. Chainbi, "A survey on energy aware VM consolidation strategies," *Sustain. Comput., Informat. Syst.*, vol. 23, pp. 80–87, Sep. 2019, doi: 10.1016/j.suscom.2019.06.003.

[4] A. H. T. Dias, L. H. A. Correia, and N. Malheiros, "A systematic literature review on virtual machine consolidation," *ACM Comput. Surv.*, vol. 54, no. 8, pp. 1–38, Nov. 2022, doi: 10.1145/3470972.

[5] A. Ashraf, B. Byholm, and I. Porres, "Distributed virtual machine consolidation: A systematic mapping study," *Comput. Sci. Rev.*, vol. 28, pp. 118–130, May 2018, doi: 10.1016/j.cosrev.2018.02.003.

[6] R. Yadav, W. Zhang, K. Li, C. Liu, and A. A. Laghari, "Managing overloaded hosts for energy-efficiency in cloud data centers," *Cluster Comput.*, vol. 24, no. 3, pp. 2001–2015, Sep. 2021, doi: 10.1007/s10586-020-03182-3.

[7] A. Beloglazov, J. Abawajy, and R. Buyya, "Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing," *Future Gener. Comput. Syst.*, vol. 28, no. 5, pp. 755–768, May 2012, doi: 10.1016/j.future.2011.04.017.

[8] F. D. Rossi, M. G. Xavier, C. A. F. De Rose, R. N. Calheiros, and R. Buyya, "E-eco: Performance-aware energy-efficient cloud data center orchestration," *J. Netw. Comput. Appl.*, vol. 78, pp. 83–96, Jan. 2017, doi: 10.1016/j.jnca.2016.10.024.

[9] R. Bradford, E. Kotsovinos, A. Feldmann, and H. Schiöberg, "Live wide-area migration of virtual machines including local persistent state," in *Proc. 3rd Int. Conf. Virtual Execution Environ.* New York, NY, USA: Association for Computing Machinery, Jun. 2007, pp. 169–179, doi: 10.1145/1254810.1254834.

[10] X. Ma, H. Xu, H. Gao, M. Bian, and W. Hussain, "Real-time virtual machine scheduling in industry IoT network: A reinforcement learning method," *IEEE Trans. Ind. Informat.*, vol. 19, no. 2, pp. 2129–2139, Feb. 2023, doi: 10.1109/TII.2022.3211622.

[11] U. Deshpande and K. Keahey, "Traffic-sensitive live migration of virtual machines," *Future Gener. Comput. Syst.*, vol. 72, pp. 118–128, Jul. 2017, doi: 10.1016/j.future.2016.05.003.

[12] M. Awad, N. Kara, and A. Leivadeas, "Utilization prediction-based VM consolidation approach," *J. Parallel Distrib. Comput.*, vol. 170, pp. 24–38, Dec. 2022, doi: 10.1016/j.jpdc.2022.08.001.

[13] B. Liang, X. Dong, Y. Wang, and X. Zhang, "A high-applicability heterogeneous cloud data centers resource management algorithm based on trusted virtual machine migration," *Expert Syst. Appl.*, vol. 197, Jul. 2022, Art. no. 116762, doi: 10.1016/j.eswa.2022.116762.

[14] F. Liu, Z. Ma, B. Wang, and W. Lin, "A virtual machine consolidation algorithm based on ant colony system and extreme learning machine for cloud data center," *IEEE Access*, vol. 8, pp. 53–67, 2020, doi: 10.1109/ACCESS.2019.2961786.

[15] A. Beloglazov and R. Buyya, "Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers," *Concurrency Comput., Pract. Exper.*, vol. 24, no. 13, pp. 1397–1420, Sep. 2012, doi: 10.1002/cpe.1867.

[16] S. Singh and R. Kumar, "Energy efficient optimization with threshold based workflow scheduling and virtual machine consolidation in cloud environment," *Wireless Pers. Commun.*, vol. 128, no. 4, pp. 2419–2440, Feb. 2023, doi: 10.1007/s11277-022-10049-w.

[17] S. Kulshrestha and S. Patel, "An efficient host overload detection algorithm for cloud data center based on exponential weighted moving average," *Int. J. Commun. Syst.*, vol. 34, no. 4, p. e4708, Mar. 2021, doi: 10.1002/dac.4708.

[18] N. K. Biswas, S. Banerjee, U. Biswas, and U. Ghosh, "An approach towards development of new linear regression prediction model for reduced energy consumption and SLA violation in the domain of green cloud computing," *Sustain. Energy Technol. Assessments*, vol. 45, Jun. 2021, Art. no. 101087, doi: 10.1016/j.seta.2021.101087.

[19] V. D. Reddy, G. R. Gangadharan, and G. S. V. R. K. Rao, "Energy-aware virtual machine allocation and selection in cloud data centers," *Soft Comput.*, vol. 23, no. 6, pp. 1917–1932, Mar. 2019, doi: 10.1007/s00500-017-2905-z.

[20] J. Ahmadi, A. T. Haghighat, A. M. Rahmani, and R. Ravanmehr, "A flexible approach for virtual machine selection in cloud data centers with AHP," *Softw., Pract. Exper.*, vol. 52, no. 5, pp. 1216–1241, May 2022, doi: 10.1002/spe.3062.

[21] N. Baskaran and R. Eswari, "Efficient VM selection strategies in cloud datacenter using fuzzy soft set," *J. Org. End User Comput.*, vol. 33, no. 5, pp. 153–179, Sep. 2021, doi: 10.4018/JOEUC.20210901.oa8.

[22] M. S. Mekala and P. Viswanathan, "Energy-efficient virtual machine selection based on resource ranking and utilization factor approach in cloud computing for IoT," *Comput. Electr. Eng.*, vol. 73, pp. 227–244, Jan. 2019, doi: 10.1016/j.compeleceng.2018.11.021.

[23] H. Li, W. Li, H. Wang, and J. Wang, "An optimization of virtual machine selection and placement by using memory content similarity for server consolidation in cloud," *Future Gener. Comput. Syst.*, vol. 84, pp. 98–107, Jul. 2018, doi: 10.1016/j.future.2018.02.026.

[24] K. Haghshenas and S. Mohammadi, "Prediction-based underutilized and destination host selection approaches for energy-efficient dynamic VM consolidation in data centers," *J. Supercomput.*, vol. 76, no. 12, pp. 10240–10257, Dec. 2020, doi: 10.1007/s11227-020-03248-4.

[25] P. Chhikara, R. Tekchandani, N. Kumar, and M. S. Obaidat, "An efficient container management scheme for resource-constrained intelligent IoT devices," *IEEE Internet Things J.*, vol. 8, no. 16, pp. 12597–12609, Aug. 2021, doi: 10.1109/JIOT.2020.3037181.

[26] C. Chen, K. He, and Q. Guan, "Minimum migration time selection algorithm for container consolidation," in *Proc. IEEE Int. Conf. Inf. Autom. (ICIA)*, Aug. 2018, pp. 1664–1668, doi: 10.1109/ICInfA.2018.8812421.

[27] W. Fan, Z. Han, P. Li, J. Zhou, J. Fan, and R. Wang, "A live migration algorithm for containers based on resource locality," *J. Signal Process. Syst.*, vol. 91, no. 10, pp. 1077–1089, Oct. 2019, doi: 10.1007/s11265-018-1401-8.

[28] J.-P. Luo, X. Li, and M.-R. Chen, "Hybrid shuffled frog leaping algorithm for energy-efficient dynamic consolidation of virtual machines in cloud data centers," *Expert Syst. Appl.*, vol. 41, no. 13, pp. 5804–5816, Oct. 2014, doi: 10.1016/j.eswa.2014.03.039.

[29] L. He, D. Zou, Z. Zhang, C. Chen, H. Jin, and S. A. Jarvis, "Developing resource consolidation frameworks for moldable virtual machines in clouds," *Future Gener. Comput. Syst.*, vol. 32, pp. 69–81, Mar. 2014, doi: 10.1016/j.future.2012.05.015.

[30] K. Bibiks, Y.-F. Hu, J.-P. Li, P. Pillai, and A. Smith, "Improved discrete cuckoo search for the resource-constrained project scheduling problem," *Appl. Soft Comput.*, vol. 69, pp. 493–503, Aug. 2018, doi: 10.1016/j.asoc.2018.04.047.

[31] A. Moazeni, R. Khorsand, and M. Ramezanpour, "Dynamic resource allocation using an adaptive multi-objective teaching-learning based optimization algorithm in cloud," *IEEE Access*, vol. 11, pp. 23407–23419, 2023, doi: 10.1109/ACCESS.2023.3247639.

[32] B. Liang, D. Wu, P. Wu, and Y. Su, "An energy-aware resource deployment algorithm for cloud data centers based on dynamic hybrid machine learning," *Knowl.-Based Syst.*, vol. 222, Jun. 2021, Art. no. 107020, doi: 10.1016/j.knosys.2021.107020.

[33] M. Rezakhani, N. Sarrafzadeh-Ghadimi, R. Entezari-Maleki, L. Sousa, and A. Movaghar, "Energy-aware QoS-based dynamic virtual machine consolidation approach based on RL and ANN," *Cluster Comput.*, Feb. 2023, doi: 10.1007/s10586-023-03983-2.
[34] S. Long, Y. Li, J. Huang, Z. Li, and Y. Li, "A review of energy efficiency evaluation technologies in cloud data centers," *Energy Buildings*, vol. 260, Apr. 2022, Art. no. 111848, doi: 10.1016/j.enbuild.2022.111848.
[35] H. Feng, Y. Deng, and J. Li, "A global-energy-aware virtual machine placement strategy for cloud data centers," *J. Syst. Archit.*, vol. 116, Jun. 2021, Art. no. 102048, doi: 10.1016/j.sysarc.2021.102048.
[36] V. Garg and B. Jindal, "Energy efficient virtual machine migration approach with SLA conservation in cloud computing," *J. Central South Univ.*, vol. 28, no. 3, pp. 760–770, Mar. 2021, doi: 10.1007/s11771-021-4643-8.
[37] M. Ranjbari and J. Akbari Torkestani, "A learning automata-based algorithm for energy and SLA efficient consolidation of virtual machines in cloud data centers," *J. Parallel Distrib. Comput.*, vol. 113, pp. 55–62, Mar. 2018, doi: 10.1016/j.jpdc.2017.10.009.
[38] Z. Li, "An adaptive overload threshold selection process using Markov decision processes of virtual machine in cloud data center," *Cluster Comput.*, vol. 22, no. S2, pp. 3821–3833, Mar. 2019, doi: 10.1007/s10586-018-2408-4.
[39] N. Najafizadegan, E. Nazemi, and V. Khajehvand, "An autonomous model for self-optimizing virtual machine selection by learning automata in cloud environment," *Softw., Pract. Exper.*, vol. 51, no. 6, pp. 1352–1386, Jun. 2021, doi: 10.1002/spe.2960.
[40] Z. Zhou, Z. Hu, and K. Li, "Virtual machine placement algorithm for both energy-awareness and SLA violation reduction in cloud data centers," *Sci. Program.*, vol. 2016, Mar. 2016, Art. no. e5612039, doi: 10.1155/2016/5612039.
[41] R. Mandal, M. K. Mondal, S. Banerjee, and U. Biswas, "An approach toward design and development of an energy-aware VM selection policy with improved SLA violation in the domain of green cloud computing," *J. Supercomput.*, vol. 76, no. 9, pp. 7374–7393, Sep. 2020, doi: 10.1007/s11227-020-03165-6.
[42] E. Arianyan, H. Taheri, and S. Sharifian, "Novel energy and SLA efficient resource management heuristics for consolidation of virtual machines in cloud data centers," *Comput. Electr. Eng.*, vol. 47, pp. 222–240, Oct. 2015, doi: 10.1016/j.compeleceng.2015.05.006.
[43] M. Ghobaei-Arani, A. A. Rahmanian, M. Shamsi, and A. Rasouli-Kenari, "A learning-based approach for virtual machine placement in cloud data centers," *Int. J. Commun. Syst.*, vol. 31, no. 8, p. e3537, May 2018, doi: 10.1002/dac.3537.
[44] M. Tarahomi and M. Izadi, "A prediction-based and power-aware virtual machine allocation algorithm in three-tier cloud data centers," *Int. J. Commun. Syst.*, vol. 32, no. 3, p. e3870, Feb. 2019, doi: 10.1002/dac.3870.
[45] H. Wang and H. Tianfield, "Energy-aware dynamic virtual machine consolidation for cloud datacenters," *IEEE Access*, vol. 6, pp. 15259–15273, 2018, doi: 10.1109/ACCESS.2018.2813541.
[46] V. Barthwal and M. M. S. Rauthan, "AntPu: A meta-heuristic approach for energy-efficient and SLA aware management of virtual machines in cloud computing," *Memetic Comput.*, vol. 13, no. 1, pp. 91–110, Mar. 2021, doi: 10.1007/s12293-020-00320-7.
[47] M. A. H. Monil and A. D. Malony, "QoS-aware virtual machine consolidation in cloud datacenter," in *Proc. IEEE Int. Conf. Cloud Eng. (ICE)*, Apr. 2017, pp. 81–87, doi: 10.1109/IC2E.2017.31.
[48] M. A. H. Monil and R. M. Rahman, "VM consolidation approach based on heuristics, fuzzy logic, and migration control," *J. Cloud Comput.*, vol. 5, no. 1, Dec. 2016, doi: 10.1186/s13677-016-0059-7.
[49] W. Li, Q. Fan, W. Cui, F. Dang, X. Zhang, and C. Dai, "Dynamic virtual machine consolidation algorithm based on balancing energy consumption and quality of service," *IEEE Access*, vol. 10, pp. 80958–80975, 2022, doi: 10.1109/ACCESS.2022.3194514.
[50] A. Rahmanian, G. Dastghaibyfard, and H. Tahayori, "Penalty-aware and cost-efficient resource management in cloud data centers," *Int. J. Commun. Syst.*, vol. 30, no. 8, p. e3179, May 2017, doi: 10.1002/dac.3179.

ZHOUJUN MA received the B.S. degree in electrical engineering from Southeast University, in 2009, and the M.S. degree in electrical engineering from Shanghai Jiao Tong University, in 2013. He is currently pursuing the Ph.D. degree with Hohai University. He is a Senior Engineer with the State Grid Jiangsu Electric Power Company, China. His current research interests include power distribution dispatching, operation and control of power systems, data center architecture, and cloud computing.

DI MA was born in 1990. He received the master's degree in power systems and automation from Nanjing, China. He has been engaged in power dispatch and monitoring work for five years. In 2019, he obtained the title of engineer and published articles, such as titled "An Improved Algorithm for AC Microgrid Line Protection Considering the Influence of Transition Resistance." His current research interests include offshore wind power, energy optimization, and low-frequency transmission.

MENGJIE LV was born in Hai'an, Jiangsu, China, in 1990. She received the master's degree in electrical engineering from the Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 2015. She joined as an Electrical Engineer for the primary design of substation with State Grid Nanjing Power Supply Company, in 2015. Her current research interests include the operation and control of power systems, data center architecture, resource management and optimization, and cloud computing.

YUTONG LIU received the B.S. degree in network engineering from the Xi'an University of Technology, in 2022. She is currently pursuing the M.S. degree with Northwestern Polytechnic University as a Non-full-time graduate student. She joined Data Center at State Grid Nanjing Power Supply Company, in 2022. Her current research interests include cloud computing, edge computing, and resource management and optimization.